

# Creating Safe State Machines

By Sam Zhong Zhang  
Technical Marketing Engineer  
Mentor Graphics Corporation

# Creating Safe State Machines

## Introduction

Finite state machines are widely used in digital circuit designs. Generally, when designing a state machine using an HDL, the synthesis tools will optimize away all states that cannot be reached and generate a highly optimized circuit. Sometimes, however, the optimization is not acceptable. For example, if the circuit powers up in an invalid state, or the circuit is in an extreme working environment and a glitch sends it into an undesired state, the circuit may never get back to its normal operating condition.

This paper discusses a general methodology when creating a state machine using the HDL Designer Series™ *State Diagram Editor*. You can specify the design so that synthesis tools will not optimize away those unused states, thus a “safe” state machine can be generated.

## “Safe” and “Unsafe” State Machines

Not all state machine designs are “unsafe.” “Safe” depends on how many states are in a design and how you define the state encoding styles:

### “Safe” State Machines

If the number of states (N) is a power of 2, and you use a binary or gray-code encoding algorithm, the state machine is “safe”. This ensures that you have M number of registers where  $N = 2^M$ . Because all of the possible state values (or register statuses) are reachable, the design is “safe.”

### “Unsafe” State Machines

If the number of states is not a power of 2, or if you do not use binary or gray-code encoding algorithm, e.g. one-hot, the state machine is “unsafe.”

For example, Figure 1 shows a design that contains 4 states:

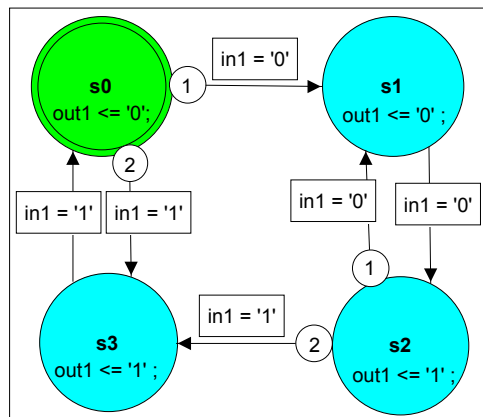


Figure 1. NRZ-to-Manchester Conversion

The number of states is 4, if you use the one-hot encoding algorithm. For example:

```
s0 => 0001
s1 => 0010
s2 => 0100
s3 => 1000
```

Thus there are 12 more states which are not defined. Generally these states are covered by the “others” (for VHDL) or “default” (for Verilog) branch of the case statement. The default operation of the synthesis tool will optimize away unreachable states in order to get a high performance circuit. But the optimization will create an “unsafe” circuit.

### Hard-encoded “Safe” State Machines:

Regardless of how many states you have, if you use the “bit-level” encoding scheme, the optimized result will be “safe.” This requires you to specify bit patterns for your states. For example, in VHDL use `std_logic_vector` to define your state type, then you can detect the undesired states using the “others” statement in the state decoding process or by explicitly defining if:

```
current_state = (undesired states) or current_state /= (desired states).
```

For the example shown in Figure 1, you can use the following statements to declare the state values:

```
SUBTYPE STATE_TYPE IS std_logic_vector(3 DOWNT0 0);  
  
CONSTANT s0 : STATE_TYPE := "0001";  
CONSTANT s1 : STATE_TYPE := "0010";  
CONSTANT s2 : STATE_TYPE := "0100";  
CONSTANT s3 : STATE_TYPE := "1000";  
  
SIGNAL current_state : STATE_TYPE ;  
SIGNAL next_state : STATE_TYPE ;
```

The example uses one-hot encoding; it also applies to any other encoding algorithm.

## Creating “Hard” Encoded “Safe” State Machines Using the State Diagram Editor

The *State Diagram Editor* of the HDL Designer Series fully supports creating hard-encoded “safe” state machines as described above. In order to achieve this result, the following options need to be set in the *State Machine Properties* dialog box:

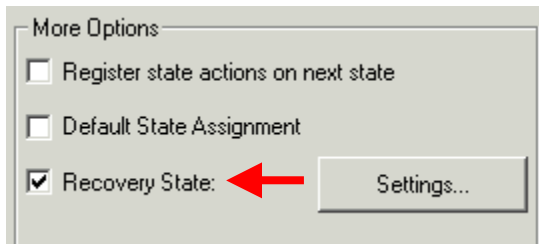


Figure 2. Menu: State Machine Properties > Generation

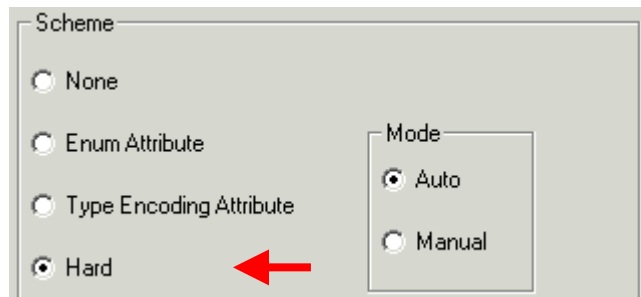


Figure 3. Menu: State Machine Properties > Encoding

The *Recovery State* can be defined as `<start_state>`, `<current_state>`, or a specific state. In the generated HDL, this will be the “others” (for VHDL) or “default” (for Verilog) branch of the case statement. The encoding mode for *Hard* can be *Auto* or *Manual*. If *Auto* is chosen, the width of `STATE_TYPE` will be calculated by using the least number of bits based on the number of states. If *Manual* is selected, you must define the value for each state in the State Diagram Editor and all the values must have the same size.

Figure 4 shows a “hard” manual encoded state machine using one-hot scheme.

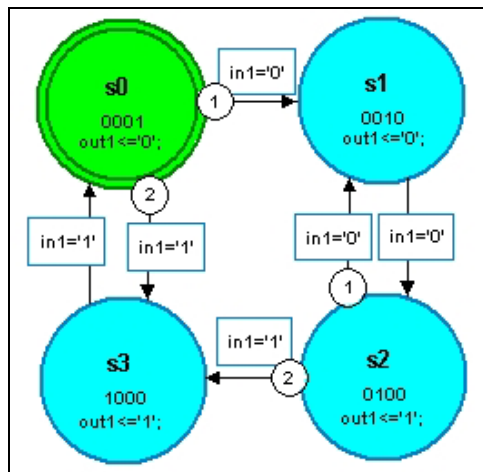


Figure 4. NRZ-to-Manchester Conversion Using Hard Manual One-Hot Encoding Scheme

## Other Information

For additional information about HDL Designer Series, please visit <http://www.hdl designer.com>, or send email to [hdl designer info@mentor.com](mailto:hdl designer info@mentor.com)

For any questions or comments on this document, please contact [hds\\_tme@mentor.com](mailto:hds_tme@mentor.com).

Copyright © 2002 Mentor Graphics Corporation. HDL Designer Series is a trademark of Mentor Graphics Corporation. All trademarks mentioned in this document are trademarks of their respective owners.

Corporate Headquarters  
Mentor Graphics  
Corporation  
8005 SW Boeckman Road  
Wilsonville, OR 97070-  
7777  
Phone: 503-685-7000

Sales and Product  
Information  
Phone: 800-547-3000  
503-685-8000

Silicon Valley  
Headquarters  
Mentor Graphics  
Corporation  
1001 Ridder Park Drive  
San Jose, California 95131  
USA

Phone: 408-436-1500  
Fax: 408-436-1501

North American Support  
Center  
Phone: 800-547-4303

Europe Headquarters  
Mentor Graphics  
Corporation  
Immeuble le Pasteur  
13/15, rue Jeanne  
Braconnier  
92360 Meudon La Forêt  
France  
Phone: 33 (0) 1-40-94-  
74-74  
Fax: 33 (0) 1-46-01-91-  
73

Pacific Rim Headquarters  
Mentor Graphics (Taiwan)  
Room 1603, 16F  
International Trade  
Building  
No. 333, Section 1, Keelung  
Road  
Taipei, Taiwan, ROC  
Phone: 886-2-87252000  
Fax: 886-2-27576027

Japan Headquarters  
Mentor Graphics Japan Co.,  
Ltd.  
Gotenyama Hills  
7-35, Kita-Shinagawa 4-  
chome  
Shinagawa-Ku, Tokyo 140  
Japan  
Phone: 81-3-5488-3030  
Fax: 81-3-5488-3021

